IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re application of:	)	Stephen R. Van Doren et al.
Serial No.:	)	10/760,640
Filed:	)	January 20, 2004
Confirmation No.:	)	9874
Group Art Unit:	)	2188
Examiner:	)	Mardochee Chery
Docket No.:	)	200313588-1
For:	)	**CACHE COHERENCY PROTOCOL WITH ORDERING POINTS**

Mail Stop Appeal Briefs - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

AMENDED APPEAL BRIEF

Sir:

In response to the Notification of Non-Compliant Appeal Brief dated September 26, 2007, and following the Notice of Appeal and the Notice of Panel Decision from Pre-Appeal Brief Review dated July 11, 2007, Appellant presents this Amended Appeal Brief.

I. TABLE OF CONTENTS

II. REAL PARTY IN INTEREST

The real party in interest is Hewlett-Packard Development Company, L.P., as indicated by the Assignment recorded January 20, 2004, Reel/Frame: 014919/0252.

III. RELATED APPEALS AND INTERFERENCES

There are no related appeals or interferences.

IV. STATUS OF CLAIMS

Claims 1-24 are pending. Claims 1-6, 11-13, 15-21 and 24 stand rejected under 35 U.S.C. §102(e) as being anticipated by U.S. Publication No. 2005/0251626 to Glasco (hereinafter, "Glasco"). Claims 7, 8, 9 and 22-23 stand rejected under 35 U.S.C. §103(a) as being obvious by Glasco in view of U.S. Patent No. 6,138,218 to Arimilli (hereinafter, "Arimilli"). Claims 10 and 14 are objected to as being dependent upon a rejected base claim, but would be allowable if rewritten in independent form including all of the limitations of the base claim and any intervening claims.

The rejection of claims 1-9, 11-13 and 15-24 is hereby being appealed.

V. STATUS OF AMENDMENTS

A Final Office Action (hereinafter, "Final Action") was issued for the present application on April 18, 2007. No amendments were made to the claims after the Final Action.

VI. SUMMARY OF THE CLAIMED SUBJECT MATTER

One aspect of the present invention, as recited in claim 1, is directed to a system (10 of FIG. 1) comprising a first node (14 of FIG. 1) having an associated cache (24 of FIG. 1; para. [0024]-[0026]). The cache (24 of FIG. 1) of the first node (14 of FIG. 1) includes data having an associated first cache state (para. [0027]). The first cache state is capable of identifying the first node (14 of FIG. 1) as being an ordering point for serializing requests from other nodes (12, 20 of FIG. 1) for the data (para. [0027]).

B. Claim 2

Claim 2 is directed to the system (10 of FIG. 1) of claim 1, wherein the first cache state enables the first node (14 of FIG. 1) to provide a data response to a request for the data from a second node (12 of FIG. 1) for the data without updating a system memory (16 of FIG. 1). The data response comprises a copy of the data requested from the second node (12 of FIG. 1; para. [0043]-[0044]).

C. Claim 3

Claim 3 is directed to the system (10 of FIG. 1) of claim 1, wherein the first cache state enables the first node (14 of FIG. 1) to provide an ownership data response to a request for the data from a second node (12 of FIG. 1; para. [0044]). The ownership data response transfers the ordering point from the first node (14 of FIG. 1) to the second node (12 of FIG. 1) the ownership data response comprising a copy of the data requested from the second node (12 of FIG. 1; para. [0043]-[0044]).

D. Claim 4

Claim 4 is directed to the system (10 of FIG. 1) of claim 3, wherein the first node (14 of FIG. 1) provides the ownership data response without updating a system memory (16 of FIG. 1; para. [0044]).

E. Claim 5

Claim 5 is directed to the system (10 of FIG. 1, 100 of FIG. 3) of claim 3, wherein the first node (14 of FIG. 1, 104 of FIG. 3) defines a first processor (14 of FIG. 1, 104 of FIG. 3) and the second node (12 of FIG. 1, 102 of FIG. 3) defines a second processor (12 of FIG. 1, 102 of FIG. 3, para. [0024]-[0025]). Each of the first processor (14 of FIG. 1, 104 of FIG. 3) and the second processor (12 of FIG. 1, 102 of FIG. 3) has an associated cache (22, 24 of FIG. 1, 104, 106, 114 of FIG. 3), the associated caches (22, 24 of FIG. 1, 104, 114 of FIG. 3) of the first and second processors (14 of FIG. 1, 104 of FIG. 3; 12 of FIG. 1, 102 of FIG. 3) each comprises a plurality of cache lines (116 of FIG. 3). Each cache line (116 of FIG. 3) has a respective tag address that identifies associated data (para. [0027], [0055]) for the respective cache line. Furthermore, each cache line (116 of FIG. 3) has state information that indicates a state of the associated data for the respective cache line (116 of FIG. 3, para. [0027]). The system (10 of FIG. 1, 100 of FIG. 3) further comprises first (14 of FIG. 1, 104 of FIG. 3) and second processors (12 of FIG. 1, 102 of FIG. 3) capable of communicating with each other and with other nodes (20 of FIG. 1, 106 of FIG. 3) of the system (10 of FIG. 1, 100 of FIG. 3) through an interconnect (18 of FIG. 1, 108 of FIG. 3; para. [0025], [0052]-[0057]).

F. Claim 6

Claim 6 is directed to the system (10 of FIG. 1, 100 of FIG. 3) of claim 5, further comprising a first cache controller (14 of FIG. 1, 104 of FIG. 3) associated with the first processor and a second cache controller (12 of FIG. 1, 118 of FIG. 3) associated with the second processor (102 of FIG. 3; para. [0056]). The first cache controller (118 of FIG. 3) is operative to manage data requests and responses for the associated cache (24 of FIG. 1, 114 of FIG. 3) of the first processor (14 of FIG. 1, 104 of FIG. 3; para. [0056]-[0057]). The first cache controller (14 of FIG. 1, 104 of FIG. 3) effects state transitions associated with the data in the associated cache (24 of FIG. 1, 114 of FIG. 3) of the first processor (14 of FIG. 1, 104 of FIG. 3) based on the data requests and responses for the associated cache (24 of FIG. 1, 114 of FIG. 3) of the first processor (104 of FIG. 3; para. [0056]). The second cache controller (12 of FIG. 1, 118 of FIG. 3) is operative to manage data requests and responses for the associated cache of the second processor (22 of FIG. 1, 114 of FIG. 3). The second cache controller (12 of FIG. 1, 102 of FIG. 3) effects state transitions associated with the data in the associated cache (22 of FIG. 1, 114 of FIG. 3) of the second processor (12 of FIG. 1, 102 of FIG. 3) based on the data requests and responses for the associated cache (22 of FIG. 1, 114 of FIG. 3) of the second processor (12 of FIG. 1, 102 of FIG. 3; para. [0056]-[0060]).

G. Claim 7

Claim 7 is directed to the system (10 of FIG. 1, 100 of FIG. 3) of claim 5, wherein the system implements a hybrid cache coherency protocol (para. [0029]). Each of the first (14 of FIG. 1, 104 of FIG. 3) and second processors (12 of FIG. 1, 102 of FIG. 3) employs a source broadcast-based protocol to issue a request for the data and employs

an associated forward progress protocol to reissue a request for the data if the request fails in the source broadcast protocol (para. [0029]).

H. Claim 8

Claim 8 is directed to the system (10 of FIG. 1, 100 of FIG. 3) of claim 7, wherein the forward progress protocol comprises a null-directory protocol (para. [0029]).

I. Claim 11

In another aspect of the present invention, as recited in claim 11, a multi-processor network (50 of FIG. 2) comprising a plurality of processor nodes (54-60 of FIG. 2), each processor node (54-60 of FIG. 2) having at least one associated cache (64-70 of FIG. 2; para. [0045]-[0046]). The plurality of processor nodes (54-60 of FIG. 2) employ a coherency protocol (para. [0046]). The coherency protocol employ ordering points for serializing requests for data associated with the at least one associated cache (64-70 of FIG. 2) of the plurality of processor nodes (54-60 of FIG. 2; para. [0051]). The ordering point for the data is identified by a cache state that is associated with the at least one associated cache (64-70 of FIG. 2) of one of the processor nodes (54-60 of FIG. 2; para. [0051]).

J. Claim 12

Claim 12 is directed to the multi-processor network (50 of FIG. 2) of claim 11, wherein the coherency protocol employs a first cache state at a first node of the plurality of processor nodes (54-60 of FIG. 2) to identify the first node as an ordering point for the data (para. [0050]-[0051]). The first cache state enables the first node (56 of FIG. 2) to provide a data response to a request from a second node (54 of FIG. 2) for the data without updating a system memory (72 of FIG. 2; para. [0051]).

K. Claim 13

Claim 13 is directed to the multi-processor network (50 of FIG. 2) of claim 11, wherein an ownership data response transfers the ordering point from the first node (56 of FIG. 2) to the second node (54 of FIG. 2; para. [0023]).

L. Claim 15

Another aspect of the present invention as recited in claim 15 is directed to a system (10 of FIG. 1, 50 of FIG. 2, 100 of FIG. 3) comprising means for employing a cached ordering point to serialize requests for a block of data from nodes (12, 14 and 20 of FIG. 1, 54-60 and 82 of FIG. 2, 102-106 of FIG. 3) of the system (10 of FIG. 1, 50 of FIG. 2, 100 of FIG. 3, (para. [0020], [0045]-[0046], [0052]-[0054]). The system (10 of FIG. 1, 50 of FIG. 2, 100 of FIG. 3) further comprising means for associating (12, 14 and 20 of FIG. 1, 54-60 and 82 of FIG. 2, 102-106 and 118 of FIG 3) the cached ordering point for the block of data with a first node of the system (10 of FIG. 1, 50 of FIG. 2, 100 of FIG. 3; (para. [0021]-[0027], [0045]-[0051], [0052]-[0060])).

M. Claim 16

Claim 16 is directed to the system (10 of FIG. 1, 50 of FIG. 2, 100 of FIG. 3) of claim 15, wherein the means for associating (12, 14 and 20 of FIG. 1, 54-60 and 82 of FIG. 2, 102-106 and 118 of FIG 3) the ordering point comprises means for assigning a first cache state to the first node of the system (14 of FIG. 1, 56 of FIG 2., 104 of FIG 3; para. [0020]-[0027], [0045]-[0051], [0052]-[0060])).

N. Claim 17

Claim 17 is directed to the system (10 of FIG. 1, 50 of FIG. 2, 100 of FIG. 3) of claim 15, further comprising means for providing a data response from the first node (14

of FIG. 1, 56 of FIG 2., 104 of FIG 3) to a request from a second node (12 of FIG. 1, 54 of FIG 2., 102 of FIG 3) for the data without updating a system memory (16 of FIG. 1, 72 of FIG. 2, 110 of FIG. 3; para. [0020]-[0027], [0045]-[0051], [0052]-[0060]).

O. Claim 18

Claim 18 is directed to the system (10 of FIG. 1, 50 of FIG. 2, 100 of FIG. 3) the system of claim 15, further comprising means for providing an ownership data response from the first node (14 of FIG. 1, 56 of FIG 2., 104 of FIG 3) to a request from a second node (12 of FIG. 1, 54 of FIG 2., 102 of FIG 3) for the data (para. [0020]-[0027], [0045]-[0051], [0052]-[0060]). The ownership data response transfers the ordering point from the first node (14 of FIG. 1, 56 of FIG 2., 104 of FIG 3) to the second node (12 of FIG. 1, 54 of FIG 2., 102 of FIG 3; para. [0020]-[0027], [0045]-[0051], [0052]-[0060]).

P. Claim 19

Yet a further aspect of the present invention as recited in claim 19 is directed to a method that includes employing a first cache state to identify a first node (14 of FIG. 1) of a system (10 of FIG. 1) as being an ordering point for a block of data (para. [0030]-[0031]). The method further providing a data response from the first node (14 of FIG. 1) to a request from a second node (12 of FIG. 1) of the system for the block of data (para. [0030]-[0031]).

Q. Claim 22

Claim 22 is directed to the method of claim 19, wherein providing a data response comprises employing a source broadcast protocol to deterministically resolve the request from the second node (12 of FIG. 1) for the block of data (para. [0030]-[0031]). The method further includes employsing a forward progress technique to

deterministically resolve the request from the second node (12 of FIG. 1) for the block of data if the request from the second node (12 of FIG. 1) cannot be deterministically resolved through employing the source broadcast protocol (para. [0029]-[0037]).

R. Claim 24

Another aspect of the present invention as recited in claim 15 is directed to a system including a coherency protocol operative to assign a cache state to a cache line (116 of FIG. 3) of one node of a plurality of nodes (102, 104, 106 of FIG. 3) of a system (100 of FIG. 3, para. [0052]-[0055]). The system (100 of FIG. 3) further including cache state defining the one node (102 of FIG. 3) as an ordering point in the system for data in the cache line (116 of FIG. 3) of the one node (102 of FIG. 3, para. [0052]-[0060]).

VII. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

A. Whether claims 1-6, 11-13, 15-21 and 24 are unpatentable under 35 U.S.C. §102(e) in view of Glasco.

B. Whether claims 7, 8, 9 and 22-23 are unpatentable under 35 U.S.C. §103 over Glasco in view of Arimilli.

VIII. ARGUMENT

A. 35 U.S.C. §102(e) rejection of claims 1-6, 11-13, 15-21 and 24 as being anticipated by Glasco

Anticipation requires that the single prior art reference disclose each and every element of the claimed invention, arranged as in the claim. *Lindemann Maschinenfabrik GmbH v. American Hoist & Derrick Co.*, 730 F.2d 1452, 1458, 221 U.S.P.Q. 481, 485 (Fed. Cir. 1984).

1. The Anticipation Rejection of Claims 1, 11, 15, 19 and 24

Glasco does not anticipate the system of claim 1.

The Final Action contends that Glasco discloses the system of claim 1 relying on para. [0045], lines 1-3; para. [0087], lines 11-16; and para. [0120]-[0123]. However, such sections do not collectively disclose each and every element of the claimed invention as arranged in claim 1. *Lindemann Maschinenfabrik GmbH v. American Hoist & Derrick Co., supra.*

In contrast to claim 1, para. [0045] of Glasco discloses that the system may include one or more memory controllers each of which operates as a serialization point for ordering data access requests. Glasco further states that each memory controller is configured to serialize requests so that only one data access request for a given memory line is allowed at any particular time. Glasco, at para. [0045]. Clearly, there is no support for claim 1 in paragraph [0045] of Glasco.

Additionally, para. [0087] of Glasco, also cited in the Final Action, discloses a coherence directory with respect to Figure 7. The coherence directory can be a full

directory or a sparse directory that includes only a limited number of entries associated with a selected set of memory lines. Glasco, at para. [0087]. Glasco teaches that the coherence directory is maintained by the memory controller (Glasco, at para. [0087]), which as set forth in para. [0045] of Glasco is a serialization point. The coherency directory thus is a mechanism within the memory controller that maintains state information for memory lines for multiple processors in the system. See Glasco at para. [0049] and [0087] and Fig. 7. However, in sharp contrast to claim 1, Glasco further teaches (consistent with the teachings in Glasco at para. [0087]-[0090]) that the home memory controller is the serialization point for a set of memory lines of a multiprocessor cluster regardless of the state of such lines. The use of the coherence directory in a multiple cluster system serves a particular purpose. According to Glasco, the coherence directory (being a serialization point for a plurality of different cache lines in different clusters) is used to reduce the number of transactions, since the coherence directory manages and filters probes that do not have to be sent to specific clusters. Glasco, at para. [0092]. Additional reference should be made to Glasco at para. [0088]-[0090] and [0092] and to the table of FIG. 7 for specific examples of how transactions are serialized at the home memory controller for different cache states and occupancy information. Since, Glasco teaches that the home memory controller is the serialization point for transactions to memory lines in the coherence directory regardless of the cache state for such memory lines, Glasco does not anticipate claim 1.

The Final Action also relies on para. [0120]-[0123] of Glasco in its rejection of claim 1. These paragraphs do not relate to or disclose any cache state that is capable of identifying the first node as being an ordering point for serializing requests from other

nodes, as recited in claim 1. Instead, para. [0120]-[0123] of Glasco relate to and describe an unrelated transaction that can be used to evict a dirty line from a cache coherence directory. However, when these paragraphs are considered in their entirety and in their intended context, as explained at para. [0124]-[0127] of Glasco, the deficiencies of Glasco relative to claim 1 become evident. Significantly, para. [0124] states that a sized write request is directed to the home memory controller for the memory line, which serializes the requests for the memory line (after determining that the memory line is a remotely cached dirty memory line) by generating probes to all local nodes. That is, it is the home memory controller, the coherence directory and the cache coherence controller (see para. [0087] of Glasco) that defines the serialization point. Additionally, Glasco explicitly teaches that the processors of a multiprocessor cluster are each coupled to the home memory controller. Glasco, at para. [0054]. As discussed above, Glasco teaches that the home memory controller is the serialization point for memory lines in a given multiprocessor cluster regardless of the cache state for the memory line. See, *e.g.*, FIG. 7 and corresponding description at para. [0087]-[0090].

In the Response to Arguments section of the Final Action, the Examiner asserts without the benefit of any legal citation that "it also suffices that the prior art discloses the claimed invention at least in the manner recited in applicant's specification." Final Action, at page 3, lines 1-3. The Examiner then proceeds to cite selected text in Appellant's specification corresponding particular examples, including features not recited in claim 1. However, it appears that the Examiner is improperly is trying to allege that if a reference (i.e., Glasco) reads on a selected portion from Appellant's

specification that the reference necessarily reads on the claims. 37 CFR 1.104(c) specifically sets forth requirements concerning the nature of an Examiner's action, in which a rejection is to made based on what is claimed. See MPEP 706. Moreover, it is well settled that, "Although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims." See *In re Van Geuns*, 988 F.2d 1181, 26 U.S.P.Q.2d 1057 (Fed. Cir. 1993). In this situation, the Examiner is attempting to contradict this settled doctrine by seeking to support the rejection of claim 1 based on vague similarities between Glasco and selected portions of Appellant's specification that include features not recited in claim 1, but describe examples of features that can be implemented.

Moreover, Appellant submits that the conclusion in the Final Action, which is made based on the comparison of Glasco (See Final Action, at page 4, numbered paragraph 4) and the selected excerpts from Appellant's specification, is contrary to the evidence presented. Significant differences between the system of claim 1 and the characterization of Glasco are readily apparent. It must be pointed out that the quoted section does not exist in Glasco as it is presented in the Final Action, but instead appears to represent a compilation of the Examiner's characterization based on various sections of Glasco. As discussed above, in harmony with the first quotation in numbered paragraph 4 of the Final Action, the system of Glasco employs a coherence directory that is contained in a memory controller to provide a serialization point. By using a coherence directory for various memory lines of a multiprocessor cluster, according to Glasco, the number of transactions can be reduced. See Glasco at para. [0049], beginning at line 6. This directory-based approach taught by Glasco is in sharp

contrast to the system of claim 1. For example, with reference to Figure 7, Glasco further explicitly discloses that the coherence directory includes state information, dirty data owner information and an occupancy vector associated with multiple memory lines. Glasco at para. [0087]. In contrast, in claim 1, the cache of a given node has data that includes a cache state that is capable of identifying the first node as an ordering point for serializing requests from other nodes for such data. Stated differently, in claim 1, it is data in the associated cache of the first node that identifies the first node as the ordering point, whereas in Glasco, and as represented by the Examiner in the Final Action, it is the entries in a coherence directory of a memory controller for multiple processors that indicate the state of the line.

Moreover, in contrast to the conclusion at page 5, line 4, of the Final Action, there is nothing in Glasco to suggest that an ordering point is transferred when a write back to memory occurs. Instead, as discussed above, the memory controller and its associated coherence directory remain the serialization point for all requests. See Glasco, at para. [0045]-[0049].

For the reasons stated above, Appellant respectfully requests that the rejection of claim 1 be withdrawn. The rejection of claims 11, 15, 19 and 24 should be withdrawn for similar reasons.

2. The Anticipation Rejection of Claim 2

Claim 2 depends from claim 1 and is not anticipated by Glasco for at least the same reasons as claim 1, and for the following reasons. In rejecting claim 2, the Final Action simply quotes claim 2 and cites para. [00131] of Glasco as the basis for the

rejection. In sharp contrast to claim 2, the cited section of Glasco at para. [0131] does not involve any data response that includes a copy of the requested data. Instead, para. [0131] of Glasco describes a validate block transaction that is issued to invalidate a memory line to achieve an eviction of the memory line that is not dirty, but clean. Since the process being described is an eviction of clean (as opposed to dirty) memory line, no data response is provided and no data is written back to memory as part of the eviction process. Glasco para. [0128], lines 1 to 4. Moreover, Glasco teaches a transaction called a "validate block" transaction to invalidate the remote copy of the cached data. See Glasco para. [0128], lines 4 to 5, and Glasco para. [0131], lines 1 to 4. The actions being described in Glasco thus differ significantly from what is recited in amended claim 2. For these reasons, Glasco fails to anticipate claim 2.

In the Response to Arguments section of the Final Action, at page 5 (paragraph b), the Examiner proffers a new position in an attempt to contradict Appellant's arguments at page 10, paragraph 3, of Appellant's Response dated January 3, 2007, which relate to claim 2. The Examiner's Response to Arguments, however, does not include further reliance on para. [0131], which is provided as the basis for the rejection of claim 2 in the Detailed Action section of the Final Action (see Final Action, at page 3). In view of the new grounds for rejecting claim 2 set forth in the Response to Arguments section (page 5 of the Final Action), it is presumed that the prior amendment to claim 2 and arguments discussed above were considered persuasive over the rejection based on para. [0131] of Glasco and that new grounds are being provided in Response to Arguments section.

The last paragraph on page 5 of the Final Action includes a single quotation allegedly from Glasco. However, the single quotation does not correspond to any single quotation that can be found in Glasco, and therefore appears to be misleading by its presentation. Regardless of its presentation, these new grounds for rejecting claim 2 set forth at page 5 of the Final Action actually support the patentability of claim 2. Significantly, the quoted text in the last paragraph at page 5 of Final Action specifically states (more than once) that the memory line must be written back to memory. In sharp contrast, claim 2 recites that the first node provides "a data response to a request from a second node **without updating a system memory…**(emphasis added)" Accordingly, there is no justification to conclude, as was done in the Final Action, that Glasco anticipates claim 2. Additionally, as discussed above with respect to Claim 1, the quoted text at page 5 of the Final Action further emphasizes the directory-based approach disclosed in Glasco, which is sharp contrast to use of a cached ordering point to serialize requests for data, as recited in claim 2 (depending from claim 1).

For the reasons stated above, Appellant respectfully requests that the rejection of claim 2 be withdrawn.

3. <u>The Anticipation Rejection of Claim 3</u>

Claim 3 depends from claim 1 and is not anticipated by Glasco for at least the same reasons as claim 1, and for the following reasons.

Respectfully, the reliance on Glasco para. [0089]-[0090] in the Final Action to reject claim 3 to reject claim 3 fails to disclose the system of claim 3. Instead, Glasco at para. [0089]-[0090] discloses different examples of how a memory controller, operating

as a serialization point, utilizes its coherence directory (FIG. 7) when a given memory line is in a modified state (para. [0089]) and an ownership state (para. [0090]), respectively. The context of para. [0089]-[0090] is to check an additional field (the dirty data owner information field) of the coherence directory for the purpose of reducing the number of transactions in the system. See Glasco, Abstract and at para. [0092]. In each of the examples relating to FIG. 7 at para. [0087]-[0092] Glasco does not teach that an ownership data response is provided that transfers an ordering point to another node, as recited in claim 3. Instead, the home memory controller is and remains the serialization point for the set of memory lines for a cluster included in the coherence directory regardless of cache state or a change in cache state. Again, this is because Glasco relates to a particular directory-based approach that employs a coherence directory in a memory controller for a multiprocessor cluster to help reduce the number of transactions. Glasco at para. [0054] and [0092].

In the Response to Arguments section, the Final Action refers back to arguments provided in the Final Action for claims 1 and 2. Any relevance of the rejection of these claims to the system of claim 3 was addressed above with respect to claims 1 and 2.

For the reasons stated above, Appellant respectfully requests that the rejection of claim 3 be withdrawn.

4. <u>The Anticipation Rejection of Claim 4</u>

Claim 4 depends from claims 1 and 3 and is not anticipated by Glasco for at least the same reasons as claims 1 and 3. Additionally, claim 4 recites that the first node provides an ownership data response without updating system memory. Accordingly,

additional reasons to support the allowance of claim 4 have been discussed above with respect to claim 2. For these reasons, Appellant respectfully requests that the rejection of claim 4 be withdrawn.

5. The Anticipation Rejection of Claim 5

Claim 5 depends from claims 1 and 3 and is not anticipated by Glasco for at least the same reasons as claims 1 and 3, and for the following reasons. The Final Action relies on para. [0059] to support a contention that each cache line has a respective tag address that identifies associated data and each cache line having state information that indicates a state of the associated data for the respective cache line. However, the reliance on para. [0059] does not include evidence to support the rejection of claim 5. In particular, the description in para. [0059] relates to information that can be found in pending buffer 309 of a protocol engine 305 that is part of a coherence controller 230. Glasco is clear at para. [0054] when it specifically teaches that the processors 202a-d are coupled to the cache coherence controller. That is, as discussed above with respect to claim 1, the cache coherence controller 230 is used as a serialization point for processors of a multiprocessor cluster.

Additionally, the Final Action contends that switch 210 shown in FIG. 2 of Glasco somehow enables communication between processors. In contrast to this contention, the switch 210 is specifically described as an I/O switch that connects the system to I/O adapters 216 and 220. See Glasco, at para. [0052], lines 16-17. Thus, the rejection of claim 5 is deficient with regard to identifying structure that enables communication between processors, as recited in claim 5.

For these reasons, Appellant respectfully requests that the rejection of claim 5 be withdrawn.

6. The Anticipation Rejection of Claim 6

Claim 6 depends from claims 1, 3 and 5 is not anticipated by Glasco for at least the same reasons as claims 1, 3 and 5 and for the specific features recited in claim 6. Appellant respectfully requests that the rejection of claim 6 be withdrawn.

B. 35 U.S.C. §103(a) Rejection of Claims 7, 8,9 and 22-23 as being Made Obvious by Glasco in view of Arimilli.

The following objective inquiry is to control the analysis under 35 U.S.C. 103:

> "Under §103, the scope and content of the prior art are to be determined; differences between the prior art and the claims at issue are to be ascertained; and the level of ordinary skill in the pertinent art resolved. Against this background the obviousness or nonobviousness of the subject matter is determined. Such secondary considerations as commercial success, longfelt but unsolved needs, failure of others, etc., might be utilized to give light to the circumstances surrounding the origin of the subject matter sought to be patented." *KSR v. Teleflex*, 550 U.S. ___, 127 S. Ct. 1727 (2007), citing *Graham v. John Deere Co. of Kansas City*, 383 U. S. 1 at 17–18 (1966).

1. The Obviousness Rejection of Claims 7, 9 and 23

Claim 7 depends from claims 1 and 5 and is patentable for at least the same reasons as discussed herein with respect to claims 1 and 5. Moreover, the addition of Arimilli does not make up for the aforementioned deficiencies of Glasco with respect to claim 1 because there is no content in the prior art that would render claim 1 obvious

and because of the significant differences between claim 1 and the teachings of Glasco in view of Arimilli.

The approach disclosed in Arimilli does not provide for each processor to reissue a request for specific data by employing a forward progress protocol if the request fails in the source broadcast protocol, as recited in claim 7. The Final Action cites the Technical Field section of Arimilli (Col. 1, lines 6-14) to support for the rejection of claim 7. However, this cited section specifically+ states that that snoop operations initiated by one device that are retried by another (*i.e.*, a different) device in a multiprocessor system. Arimilli, at Col. 1 lines 7-11. Moreover, the cited section does not teach or suggest that the retry that is performed by the different device employs a forward progress protocol. Instead, Arimilli teaches that the invention of Arimilli is concerned with making forward progress towards an ultimate state, which does not imply the use of any forward progress protocol, as appears has been inferred in the Final Action. For instance, forward progress can be achieved in many situations, including those shown and described in Arimilli, without the use of a forward progress protocol as this would be understood to one of ordinary skill in the art.

In the Response to Arguments section at page 6 of the Final Action (numbered paragraph e), the Examiner indicates that he was unable to find the teachings described in Appellant's response dated January 3, 2007. The following discussion expands on the teachings of Arimilli at Col. 5, lines 30-51, to demonstrate the differences between claim 7 and the teachings of Glasco in view of Arimilli. In the context of FIGS. 2 and 2A-2C, Arimilli teaches that "a first device such as L3 cache 118 initiates a read or rwitm operation 202 on system bus 122." Arimilli, at Col. 5, lines 30-31. The L2 cache 114

(which holds the requested item in a modified or recent state) asserts an intervention response 204 for the purpose of sourcing the data to L3 cache 118 (the requesting device). Arimilli, at Col. 5, lines 34-36. Another device, L2 cache 116 asserts a retry that impedes the intervention response 204 - corresponding to a failed intervention. Arimilli, at Col. 5, lines 36-38. The next paragraph in Col. 5 of Arimilli, beginning at line 42, teaches that when the original read or rwitm transaction is initiated and responded to by the intervention response 204 from L2 cache 114 and a retry is provided by L2 cache 116, the L2 cache 114 (corresponding to the responding cache) initiates action to alter the coherency state of the requested item 208 in its own memory or to push the item 208 to system memory. Thus, in contrast to claim 7, Arimilli teaches that it is the action initiated by the L2 cache (the responding cache) - and not the requesting cache - that provides the mechanism for making forward progress. This is further explained in fairly straightforward manner in Arimilli, beginning at Col. 6, line 39.

Moreover, Applicant submits that Arimilli further suggests that the forward progress mechanism being described is not part of a hybrid cache coherency protocol that includes both a source broadcast-based protocol and an associated forward progress protocol. For instance, beginning at Col. 7, line 15, Arimilli states that "Depending on the coherency protocol [i.e., singular] supported and design preferences…" Since neither Glasco nor Arimilli teaches or suggests the use of a hybrid cache coherency protocol as recited in claim 7, claim 7 is not made obvious by Glasco in view of Arimilli.

For these reasons, Appellant respectfully requests that the rejection of claim 7 be withdrawn. The rejection of claims 9 and 23 should be withdrawn for similar reasons and for the specific features recited in such claims.

2. The Obviousness Rejection of Claim 8

Claim 8 depends from claims 1, 5 and 7 and is patentable for at least the same reasons as discussed above with respect to claims 1, 5 and 7 and for the following reasons.

In addition to the reasons discussed in support of claim 7, Glasco in view of Arimilli fails to teach or suggest claim 8. Specifically, Arimilli fails to teach or suggest that the mechanism used to achieve forward progress corresponds to a null directory protocol, as recited in claim 8. Instead, Arimilli teaches a forward progress mechanism that can be used as part of or in conjunction with a given protocol to achieve forward progress in response to detecting an operation on the bus that was subject to a previously failed intervention. Arimilli, at Col. 6, lines 39-43. Since the combination of Glasco and Arimilli fail to teach or suggest claim 8, claim 8 is not made obvious by Glasco in view of Arimilli.

For these reasons, Appellant respectfully requests that the rejection of claim 8 be withdrawn.

IX. APPENDICES

The first attached Appendix contains a copy of the claims on appeal.

The second and third Appendices have been included to comply with statutory requirements.

No additional fees should be due for this Brief. In the event any fees are due in connection with the filing of this document, the Commissioner is authorized to charge those fees to Deposit Account No. 08-2025.

Respectfully submitted,

/Gary J Pitzer/

Gary J. Pitzer
Registration No. 39,334
Attorney for Applicant(s)

I hereby certify that this correspondence is being transmitted to the U.S. Patent and Trademark Office via electronic filing on October 12, 2007.

CUSTOMER NO.: 022879

Hewlett-Packard Company
Legal Department MS 79
3404 E. Harmony Road
Ft. Collins, CO 80528

Claims Appendix

Claim 1. (Original) A system comprising:

a first node having an associated cache including data having an associated first cache state, the first cache state being capable of identifying the first node as being an ordering point for serializing requests from other nodes for the data.

Claim 2. (Previously Presented) The system of claim 1, wherein the first cache state enables the first node to provide a data response to a request for the data from a second node for the data without updating a system memory, the data response comprising a copy of the data requested from the second node.

Claim 3. (Previously Presented) The system of claim 1, wherein the first cache state enables the first node to provide an ownership data response to a request for the data from a second node, the ownership data response transferring the ordering point from the first node to the second node the ownership data response comprising a copy of the data requested from the second node.

Claim 4. (Original) The system of claim 3, wherein the first node provides the ownership data response without updating a system memory.

Claim 5. (Original) The system of claim 3, wherein the first node defines a first processor and the second node defines a second processor, each of the first processor and the second processor having an associated cache, the associated caches of the

first and second processors each comprising a plurality of cache lines, each cache line having a respective tag address that identifies associated data and each cache line having state information that indicates a state of the associated data for the respective cache line, the first and second processors being capable of communicating with each other and with other nodes of the system through an interconnect.

Claim 6. (Original) The system of claim 5, further comprising a first cache controller associated with the first processor and a second cache controller associated with the second processor, the first cache controller being operative to manage data requests and responses for the associated cache of the first processor, the first cache controller effecting state transitions associated with the data in the associated cache of the first processor based on the data requests and responses for the associated cache of the first processor, the second cache controller being operative to manage data requests and responses for the associated cache of the second processor, the second cache controller effecting state transitions associated with the data in the associated cache of the second processor based on the data requests and responses for the associated cache of the second processor.

Claim 7. (Original) The system of claim 5, wherein the system implements a hybrid cache coherency protocol wherein each of the first and second processors employs a source broadcast-based protocol to issue a request for the data and employs an associated forward progress protocol to reissue a request for the data if the request fails in the source broadcast protocol.

Claim 8. (Original) The system of claim 7, wherein the forward progress protocol comprises a null-directory protocol.

Claim 9. (Original) The system of claim 7, wherein the source broadcast protocol comprises an incomplete protocol.

Claim 10. (Previously Presented) The system of claim 1, wherein the first node comprises a cache including a plurality of cache lines, the system being capable of assigning a cache state to each of the cache lines to identify the status of data cached in the cache line, the cache state being selected from the group consisting of:

a cache state indicating that the data is not cached in the cache line;

a cache state indicating that the data cached in the cache line is valid and unmodified, that other nodes may have valid cached copies of the data, and that the node associated with the cache line cannot respond to snoops by returning a copy of the data;

a cache state indicating that the data cached in the cache line is valid and unmodified, that the data cached in that cache line is the only cached copy of the data in the system, and that the node associated with the cache line can respond to snoops by returning a copy of the data;

a cache state indicating that the data cached in the cache line is valid and unmodified, that other nodes may have valid copies of the data, and that the node associated with the cache line can respond to snoops by returning a copy of the data;

a cache state indicating that the data cached in the cache line is valid and more up-to-date than a copy of the data stored in a system memory, that the data cached in the cache line has not been modified by the node associated with the cache line, that the data cached in the cache line is the only cached copy of the data in the system, that the node associated with the cache line can respond to snoops by returning a copy of the data, and that the node associated with the cache line writes the data back to memory upon displacement;

a cache state indicating that the data cached in the cache line is valid and has been modified, that the data cached in the cache line is the only cached copy of the data in the system, that the node associated with the cache line can respond to snoops by returning the data, and that the node associated with the cache line writes the data back to memory upon displacement;

a cache state indicating that the data cached in the cache line is valid and more up-to-date than the copy of the data stored in system memory, that the node associated with the cache line cannot modify the data cached in the cache line, that other nodes may have valid copies of the data in the cache line, that the node associated with the cache line can respond to snoops by returning the data, and that the node associated with the cache line writes the data back to memory upon displacement; and

a cache state indicating that the cache line is transitioning between cache states.

Claim 11. (Previously Presented) A multi-processor network comprising:

a plurality of processor nodes, each processor node having at least one associated cache;

the plurality of processor nodes employing a coherency protocol, the coherency protocol employing ordering points for serializing requests for data associated with the at least one associated cache of the plurality of processor nodes, the ordering point for the data being identified by a cache state that is associated with the at least one associated cache of one of the processor nodes.

Claim 12. (Original) The multi-processor network of claim 11, wherein the coherency protocol employs a first cache state at a first node of the plurality of processor nodes to identify the first node as an ordering point for the data, the first cache state enabling the first node to provide a data response to a request from a second node for the data without updating a system memory.

Claim 13. (Previously Amended) The multi-processor network of claim 11, wherein an ownership data response transfers the ordering point from the first node to the second node.

Claim 14. (Previously Amended) The multi-processor network of claim 11, the multi-processor network being capable of assigning a cache state to each associated cache of each processor node to identify the status of a block of data in the associated cache, the cache state being selected from the group consisting of:

a cache state indicating that the block of data does not exist in the associated cache;

a cache state indicating that the block of data in the associated cache is valid and unmodified, that other processor nodes may have valid copies of the block of data, and that the processor node associated with the associated cache cannot respond to snoops by returning a copy of the block of data;

a cache state indicating that the block of data in the associated cache is valid and unmodified, that the block of data in the associated cache is the only cached copy of the block of data in the multi-processor network, and that the processor node associated with the associated cache can respond to snoops by returning a copy of the block of data;

a cache state indicating that the block of data in the associated cache is valid and unmodified, that other nodes may have valid copies of the block of data, and that the processor node associated with the associated cache can respond to snoops by returning a copy of the block of data;

a cache state indicating that the block of data in the associated cache is valid and more up-to-date than a copy of the block of data stored in a system memory, that the block of data in the associated cache has not been modified, that the block of data in the associated cache is the only cached copy of the block of data in the system, and that the processor node associated with the associated cache can respond to snoops by returning a copy of the block of data and writes the block of data back to memory upon displacement;

a cache state indicating that the block of data in the associated cache is valid and has been modified, that the block of data in the associated cache is the only cached copy of the block of data in the system, and that the processor node associated with the

associated cache can respond to snoops by returning a copy of the block of data and writes the block of data back to memory upon displacement;

a cache state indicating that the block of data in the associated cache is valid and more up-to-date than the copy of the block of data stored in system memory, that the processor node associated with the associated cache cannot modify the block of data, that other processor nodes may have valid copies of the block of data, and that the processor node associated with the associated cache can respond to snoops by returning a copy of the block of data and writes the block of data back to memory upon displacement; and

a cache state indicating that the associated cache is transitioning between cache states.

Claim 15. (Original) A system comprising:

means for employing a cached ordering point to serialize requests for a block of data from nodes of the system; and

means for associating the cached ordering point for the block of data with a first node of the system.

Claim 16. (Original) The system of claim 15, wherein the means for associating the ordering point comprises means for assigning a first cache state to the first node of the system.

Claim 17. (Original) The system of claim 15, further comprising means for providing a data response from the first node to a request from a second node for the data without updating a system memory.

Claim 18. (Original) The system of claim 15, further comprising means for providing an ownership data response from the first node to a request from a second node for the data, the ownership data response transferring the ordering point from the first node to the second node.

Claim 19. (Original) A method comprising:

employing a first cache state to identify a first node of a system as being an ordering point for a block of data; and

providing a data response from the first node to a request from a second node of the system for the block of data.

Claim 20. (Original) The method of claim 19, further comprising enabling the ordering point to provide the data response without updating a system memory.

Claim 21. (Original) The method of claim 19, wherein providing a data response comprises:

providing an ownership data response; and

transferring the ordering point from the first node to the second node.

Claim 22. (Original) The method of claim 19, wherein providing a data response comprises:

employing a source broadcast protocol to deterministically resolve the request from the second node for the block of data; and

employing a forward progress technique to deterministically resolve the request from the second node for the block of data if the request from the second node cannot be deterministically resolved through employing the source broadcast protocol.

Claim 23. (Original) The method of claim 22, wherein employing a forward progress technique comprises employing a forward progress protocol.

Claim 24. (Original) A coherency protocol operative to assign a cache state to a cache line of one node of a plurality of nodes of a system, the cache state defining the one node as an ordering point in the system for data in the cache line of the one node.

Evidence Appendix

None

Related Proceedings Appendix

None